Student Name:

Student Number:

Midterm Examination
308-304B 2002: Object-oriented Design

Examiner: Prof. Hans Vangheluwe — Wednesday, February 20th, 2002

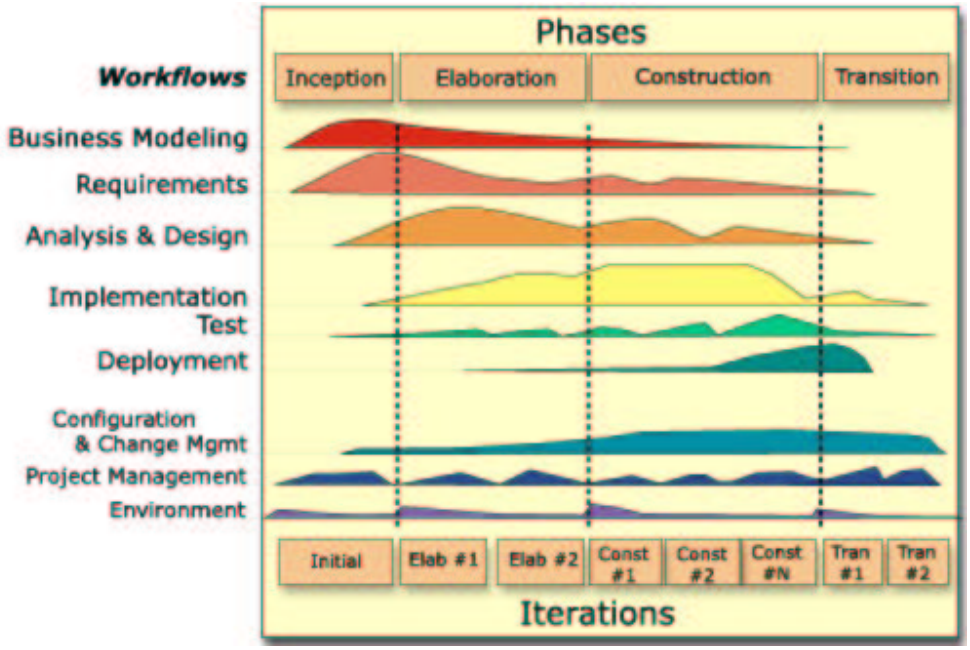
Invigilators: Jean-Sébastien Bolduc, Hesheng Chen, Ernesto Posse — 14:30 – 16:00

INSTRUCTIONS:

1. Answer all questions directly on the examination paper.
2. No aids of whatever type are permitted.
3. The exam has 14 questions on 11 pages (including cover page).
4. Attempt all questions: partial marks are given for incomplete but correct answers.
5. Numbers between brackets [] denote the weight of each question. The total is 45 points.

Good luck !

(1) [2]



What can we learn from the above figure of a software process ?

(2) [3]

- What are the characteristics of good unit tests ?

- Suppose we want to test the class `Vector` which encapsulates a 2D vector (x, y). The class has the following public methods with the obvious meanings:
  - `setX(x:Real)`
  - `setY(y:Real)`
  - `getX() -> Real`
  - `getY() -> Real`
  - `getR() -> Real`

- `getTheta() -> Real`

(r, θ) are the polar coordinates corresponding to (x, y). For the above, give a small example of each different type of test which needs to be done.

(3) [12]

Give the 9 generally accepted features of Object-Oriented systems with a short explanation for each. Give an example of the UML notation where applicable.

1.

2.

3.

4.

5.

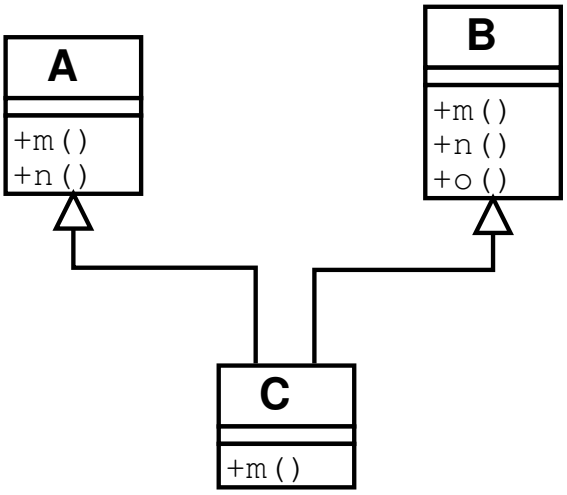
6.

7.

8.

9.

(4) [4]



1. Referring to the above figure, explain the problems with multiple inheritance and what are the alternative interpretations/implementations.

2. `a, b, c` are instances of `A, B, C` respectively. If the multiple inheritance semantics of Python is used for `class C(A,B)`, which class' method (circle the appropriate one) will be invoked with

(a) `a.m()`	A	B	C	none
(b) `a.n()`	A	B	C	none
(c) `a.o()`	A	B	C	none
(d) `b.m()`	A	B	C	none
(e) `b.n()`	A	B	C	none
(f) `b.o()`	A	B	C	none
(g) `c.m()`	A	B	C	none
(h) `c.n()`	A	B	C	none

(i) `c.o()` A B C none

3. What does one call the relationship between method `m()` in classes `A` and `C` ?

(5) [3]

- Draw in UML notation, examples of both *inheritance* and *composition*.

- How would you check, during design, whether a relationship is an inheritance relationship or not ?

(6) [3]

1. Draw a UML class diagram for a class `Object` with three subclasses `Number, String, Tuple`.

2. Add a polymorphic method `show()` to the above drawing. Explain polymorphism briefly.

3. What extra information could you add to the diagram if the subclassing were *complete* and *disjoint* ?

(7) [2]

We wish to design another class for 2D vectors (x, y). We want to implement two types of multiplication defined as follows:

- multiplication of a Vector with a scalar.

$$(x, y) * s = (x * s, y * s)$$

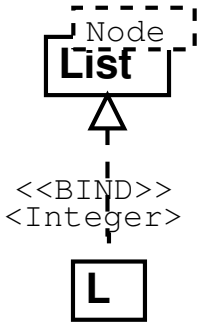
- multiplication of two Vectors

$$(x1, y1) * (x2, y2) = x1 * x2 + y1 * y2$$

- What does one call the $*$ operator in this case ?

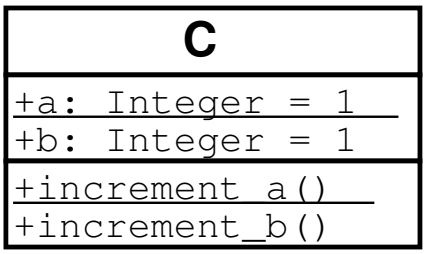
- How would this be implemented in a dynamically typed language such as Python ?

(8) [1]



Explain the above figure.

(9) [1]



For the class definition above, and after (using Python syntax)

```
c1=C()
c2=C()
c1.increment_a()
c1.increment_b()
c2.increment_a()
c2.increment_b()
```

give the values of

- `c1.a`
- `c1.b`
- `c2.a`
- `c2.b`

(10) [7]

Draw the *class diagram* for the following: A Vehicle can be specialized in two ways: both Car and Bicycle are Vehicles. A Bicycle can be owned by exactly one Person. A person has an age and a name. Man and Woman are the only possible types of Persons. A Person can own 0 or more Bicycles. A Car can be owned by 1 or two Persons. A Person can own 0 or more Cars. At any point in time, a Car can have 0 or more Persons as passengers. A car has a colour and a brand_name. Furthermore, a car always has exactly two Headlights and exactly one Engine. Each headlight can be turned on and off by sending setOn() and setOff() messages to it. The Engine can be turned on by sending it a start() message. A Car must be able to reference its Engine, but not the other way around. It must be possible for a Car object to determine its owner(s) well as its passenger(s).

(11) [1]

Explain “cascading delete” of an object.

(12) [3]

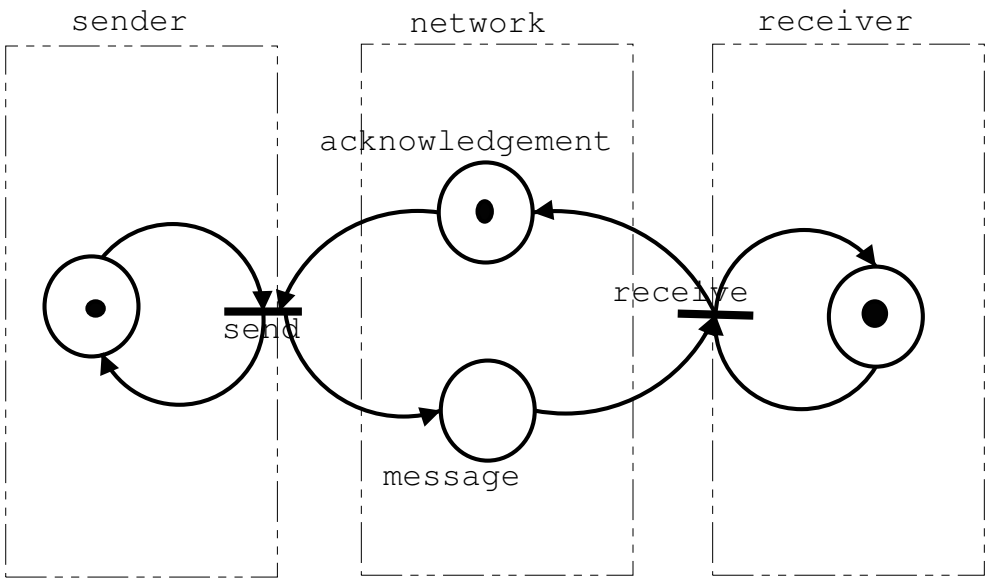
Draw both a collaboration diagram and a sequence diagram for two `Client` objects `client1` (first) and `client2` (later) registering themselves with an appropriate message with a `Server` object server. Some time later, the server will *callback* both clients and invoke their `do_it()` methods. While doing so, the server will

make it possible for the client to know *which object* called its `do_it()` method.

(13) [2]

Draw a State Automaton which recognizes the regular expression `^(ABC)+(C|D)D*$`

(14) [1]



The figure above models a simple communication between a sender and a receiver. It is assumed messages are not lost nor damaged. A token in the sender place means the sender is ready to send. A token in the receiver place means the receiver is ready to receive. A token in the acknowledgement place means there is an acknowledgement (of receipt) message on the network. A token in the message place means there is a data message on the network.

Prove that the simple protocol modelled in this network can never have two messages (a data message and an acknowledgement) simultaneously on the network.